Freie Universität Berlin

Volker Roth

Rechnersicherheit, SoSe 21

Übung 03

TutorIn: Oliver Wiese
Tutorium 02
Materialien: Latex, VSC, Skript

14. Juli 2021

---

1 Take-grant protection model

2 Project: Access control

Our project contains sensitive files, e.g. password table, log files and chat history. In this exercise, we like to improve our chat program and to study access control in practice. First, we do a code review of our own project (or if your project does not include attachments, a reviewed project of our peers). Later we will try to improve this project. In your client users can upload files and we are now interested if they can overwrite files.

(a) *Can a user overwrite a previous attachment? If so, give an example (e.g. the message or a function with the message). If not, explain why not.*

- Yes, it's possible. Because a file that a user has sent to the server is automatically downloaded by all other users.
- Also the file retains its original name and overwrites any existing files.

```
53 #handle the client commands for incoming msgs
54 def handle_msg_incoming(data):
55     command, option, msg = Header().return_header(data).get_information()
56
57     if command == send_file_command:
58
59         #open file and save data
60         with open(option, 'w') as file:
61             file.write(msg)
62         print(f'System: Received file:{option}')
```

- We fixed it by testing beforehand whether a file with the name already exists.

```
53 #handle the client commands for incoming msgs
54 def handle_msg_incoming(data):
55     command, option, msg = Header().return_header(data).get_information()

57     if command == send_file_command:
58         #TODO change msg later.
59         # option is the path
60         if os.path.isfile(option):
61             print(f'Another client tried to overwrite file {option}. File was not
       saved!')

63         #open file and save data
64         with open(option, 'w') as file:
65             file.write(msg)
66         print(f'System: Received file:{option}')
```

(b) *Can a user overwrite the log file? If so, give an example (e.g. the message or a function with the message). If not, explain why not.*

- Not possible because sent files are not stored on the servers file system. (But only in the cache)
- But the log files are on the server. So there is most likely no way to overwrite them.

(c) *Can a user overwrite the password file? If so, give an example (e.g. the message or a function with the message). If not, explain why not.*

- Not possible because sent files are not stored on the servers file system. (But only in the cache)
- But the password file is on the server. So there is most likely no way to overwrite it.

(d) *Discuss possible counter measurements against these attacks. What can the developer do to prevent such attacks? Can the operating system prevent these attacks? If so, how should be the program look like?*

- There are multiple possibilities. Either you bypass the possibility of such an attack as we just did. By not allowing the possibility at all.
- Or you theoretically allow the possibility, but add different roles and policies through the os or code for the users.
- You could implement that in your code yourself.
- Or regulate it through the OS. For example, you could only request a few or special 'system rights' from the OS, with the client program. So that the client program has no rights to overwrite a existing file.

(e) *Improve your code/docker file to prevent such attacks.*

- Done.